基于粒计算的数据分片算法的问题发现

于 磊[1,2]，罗 谦[2]，张林林[1]
（1. 西南交通大学 信息科学与技术学院，四川 成都 610031；
2. 中国民用航空局第二研究所，四川 成都 610041）

摘 要：基于粒计算的数据分片算法在考虑数据本地化的基础上，利用相似粒来划分全集数据。通过分析粒化算法的合并规则，发现在分布式数据库（Distributed Database，DDB）中，如果不考虑问题数据的出现，将会导致粒化算法产生的数据分片结果不可用，所以必须要消除问题数据。文中提出了利用黄金分割的知识进行校验，降低问题数据的产生，具有一定的可行性。文中指出如果无法保证数据的准确性，就无法使用该算法，并详细分析该算法的实际效用，对该算法的应用做了一定的限制。

关键词：相似粒；数据分片；问题数据；黄金分割
中图分类号：TP311.133.1 **文献标识码**：A **文章编号**：1673-629X(2011)06-0032-04

Problem-Finding of Data Fragment Algorithm Based on Granular Computing

YU Lei[1,2], LUO Qian[2], ZHANG Lin-lin[1]
(1. School of Information Science and Technology, Southwest Jiaotong University, Chengdu 610031, China;
2. The Second Research Institute of CAAC, Chengdu 610041, China)

Abstract: Considering data localization, the data fragment algorithm based upon granular computing uses similar granular to divide the global data. After analyzing the combination rule of the granulation algorithm, data fragment result produced by the granulation algorithm is unusable if the problem data is not considered in the distributed database. Therefore, the problem data should be eliminated. In this paper, golden section is used for detection to reduce problem data which is feasible. Last, this algorithm can not be used if data is not precise, and then practical effectiveness of it is analyzed to limit its application.

Key words: similar granulation; data fragment; data problem; golden section

0 引 言

在一个分布式数据库中如何进行合理的数据分片，直接影响到整个数据库系统的可用性、可靠性及效率等各个方面。另外，分布式数据库中分布式查询处理问题也与数据分片交织在了一起，如果数据分片不合理，将会增大远距离数据访问，使分布连接过于频繁，直接降低数据处理的性能。因此，设计出一种合理的数据分片方案就显得尤为重要。

事实上，各国的数据库方面的专家也做了大量的研究工作[1,2]，并且也得出了一定的可行方案。文献[3]提出了一种基于粒计算的数据分片算法，该算法在优先考虑数据本地化的基础上，尽可能地减少事务的数量，将相近的事务进行合并，动态地调整全集划分的粗细，将全集划分的数据分片的数量调整到了比较合理的状态。

文中在考虑到文献[3]的事务合并规则之后，发现了该算法在实际应用时存在一些问题，比如，如果数据库中某一单元数据出现异常，可能会导致最后的分片结果不可用。针对这种问题文中也提出了一种应对方案，通过理论和实例论证，表明该方案对解决数据异常问题有一定的效能，另外文中通过论述表明，如无法保证数据安全，本算法将没有实际的应用价值。

1 有关概念理解

文中只介绍与算法相关的几个基本概念，关于粗糙集的有关概念和性质可参见文献[4~6]。

定义1 数据分片[7]（Data Fragmentation）也称数据分割，是分布式数据库（DDB）的特征之一，是将分布

收稿日期：2010-11-02；**修回日期**：2011-02-20
基金项目：中国民用航空局科研项目（MHRD200924）
作者简介：于 磊（1985-），男，硕士研究生，研究方向为粒计算、人工智能。

式数据库的全集划分成相应的逻辑片段,每个逻辑片段又可由多个单元数据(存取数据的最基本单位)构成。

定义2 相似粒。令ψ和φ是信息系统S中任意两个Rough逻辑公式,$m(\varphi \wedge \psi)$和$m(\varphi \vee \psi)$是信息系统S中的两个粒,定义

$$t = |m(\varphi \wedge \psi)| / |m(\varphi \vee \psi)| \quad (1)$$

为相似度,定义当$t > 0.5$时,$m(\varphi)$与$m(\psi)$为相似粒。

定义3 粒化精度。令n_1为某一全集划分的单元数据分片总量,n_2为相近事务合并后全集划分的分片数量,则可定义粒化精度

$$\delta = n_2/n_1 (1 \geqslant \delta \geqslant 0) \quad (2)$$

粒化精度δ越大,全集划分得越细,反之全集划分得越粗。

2 基于粒计算的数据分片算法

找出相近事务并进行合并是数据粒化算法的关键,所以文献[3]就是通过运算得到相似度t,如果$t > 0.5$,就表示两者满足要求为相近事务,将其进行合并。再通过粒化精度判断合并后的结果,如果比合并前,全集划分得更粗,粒数量减少,说明将这两个相近事务进行合并是合理的,并且使得全集U划分的数据分片的数量更符合实际应用。

现在看一个例子,全集的数据集合为$D = \{r_1, r_2, \cdots, r_{10}\}$,有一组事务$T = \{t_1, t_2, t_3, t_4\}$。这里做一个规定,$tr_{ij}$表示事务$t_j$访问单元数据$r_i$所对应的值。该实例中事务访问各单元数据的情况,见表1。

表1　事务访问单元数据情况表

单元数据	事务			
	t_1	t_2	t_3	t_4
r_1	1	1	0	0
r_2	0	1	1	0
r_3	1	1	1	1
r_4	1	0	1	0
r_5	1	1	0	0
r_6	1	0	0	0
r_7	1	1	1	1
r_8	1	0	1	1
r_9	1	0	0	0
r_{10}	0	1	1	1

如果按照传统的粒化方式将全集U进行划分,会得到七个粒,$\{\{r_3, r_7\}, \{r_1, r_5\}, \{r_6, r_9\}, r_2, r_4, r_8, r_{10}\}$。

利用文献[3]中提及到的方法,如上表中,令$\psi = (t_3, 1)$,$\varphi = (t_4, 1)$,就有$m(\varphi \wedge \psi) = \{r_3, r_7, r_8, r_{10}\}$,$m(\varphi \vee \psi) = \{r_2, r_3, r_4, r_7, r_8, r_{10}\}$,通过计算得到$t = |m(\varphi \wedge \psi)| / |m(\varphi \vee \psi)| = 4/6 > 0.5$,则$m(\varphi \wedge \psi)$与$m(\varphi \vee \psi)$为相似粒。从而有$t_3$和$t_4$为相近事务,可合并$t_{34} = t_3 \vee t_4$,如$t_3$的值为1,或$t_4$的值为1,则$t_{34}$的值为1,如$t_3$、$t_4$的值均为0时,则$t_{34}$的值为0。

通过判断把相似事务t_3与t_4进行合并得到t_{34},(具体情况参见表2)。重令$B = \{t_1, t_2, t_{34}\}$,则有$U/R(B) = \{\{r_3, r_7\}, \{r_6, r_9\}, \{r_1, r_5\}, \{r_4, r_8\}, \{r_2, r_{10}\}\}$,将全集划分为5个粒,已知按照原方法划分为7个粒。这样,$\delta = 5/7$符合要求。继续判断得到t_{34}与t_2不为相近事务,同样可以得出t_{34}与t_1,t_1与t_2都不为相近事务,得到表2。

表2　合并后事务访问单元数据情况表

单元数据	事务		
	t_1	t_2	t_{34}
r_1	1	1	0
r_2	0	1	1
r_3	1	1	1
r_4	1	0	1
r_5	1	1	0
r_6	1	0	0
r_7	1	1	1
r_8	1	0	1
r_9	1	0	0
r_{10}	0	1	1

通过上面讨论,根据该思想提出了数据粒化算法。具体内容见文献[3]。

3 有关粒化算法的问题发现

文献[3]在相似粒上主要是依靠计算$|m(\varphi \wedge \psi)|$和$|m(\varphi \vee \psi)|$,但是该算法有不严密之处,由于$m(\varphi \wedge \psi) = m(\varphi) \cap m(\psi)$,$m(\varphi \vee \psi) = m(\varphi) \cup m(\psi)$,可以看出$|m(\varphi \wedge \psi)|$和$|m(\varphi \vee \psi)|$的比值$t$,主要取决于不同的值时所得的数目的比值。把不同的值的情况作为一类,重新得到一个合并规则,见表3。

表3　归结后的合并规则

$m(\psi)$	$m(\varphi)$	$m(\varphi \wedge \psi)$	$m(\varphi \vee \psi)$
0	0	0	0
0(1)	1(0)	0	1
1	1	1	1

由于在一个数据库中需要存储处理大量的数据,在存储过程中经常会发生由于恶意访问或其它的原因导致某一单元数据发生了异常[8],这种异常的发生,

会对该文献[3]算法的数据分片结果产生不同程度上的影响,这里可以把该情况发生时产生的后果归结为两类。

第一种情况,当 $m(\psi)$ 和 $m(\varphi)$ 的值相同时,无论 $m(\psi)$ 和 $m(\varphi)$ 中的哪一个值发生了改变,结果使得 t 的值变小,相似事务的数目变少,这样的情况将使全集划分得更细,系统的复杂度升高,网络流量增加,也不会对以后的应用产生大的影响。

另一种情况就是,当 $m(\psi)$ 和 $m(\varphi)$ 的值不同,不妨设 $m(\psi)$ 为 1,$m(\varphi)$ 为 0,如果数据在存储过程中出现异常,$m(\psi)$ 和 $m(\varphi)$ 中的任何一个值发生了改变,都将影响到全集的划分,将会产生较大的误差,也可导致全集划分的失败。所以,文中只讨论当 $m(\psi)$ 和 $m(\varphi)$ 的值不同时发生异常的情况。下面看一个实例。

如表 1 所示,若 tr_{42} 出现了异常 0→1,判断出 t_3 和 t_4 为相近事务,合并得到表 4。

表 4 合并后事务访问异常单元数据情况表

单元数据	事务		
	t_1	t_2	t_{34}
r_1	1	1	0
r_2	0	1	1
r_3	1	1	1
r_4	1	1	1
r_5	1	1	0
r_6	1	0	0
r_7	1	1	1
r_8	1	0	1
r_9	1	0	0
r_{10}	0	1	1

表 5 合并后事务访问异常单元数据情况表

单元数据	事务	
	t_1	t_{234}
r_1	1	1
r_2	0	1
r_3	1	1
r_4	1	1
r_5	1	1
r_6	1	0
r_7	1	1
r_8	1	1
r_9	1	0
r_{10}	0	1

重令 $B=\{t_1,t_2,t_{34}\}$,则有 $U/R(B)=\{\{r_3,r_4,r_7\},\{r_6,r_9\},\{r_1,r_5\},\{r_8\},\{r_2,r_{10}\}\}$。当按照原始的数据划分为 7 个粒,这样得到 $\delta=5/7$,将 t_3 和 t_4 进行合并。

近一步判断 t_{34} 与 t_2,从而有 t_{34} 和 t_2 为相近事务,可合并 $t_{234}=t_{34}\vee t_2$,可以得到表 5。

重令 $B=\{t_1,t_{234}\}$,则 $U/R(B)=\{\{r_1,r_3,r_4,r_5,r_7,r_8\},\{r_6,r_9\},\{r_2,r_{10}\}\}$,这样得到粒化精度 $\delta=3/5$ 符合要求,也可知将 t_{34} 和 t_2 进行了合并是合理的。

同样在令 $\psi=(t_{234},1)$,$\varphi=(t_1,1)$ 得到 t_{234} 和 t_1 为相近事务,合并得到表 6。

表 6 最终合并表

事务	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}
单元数据	1	1	1	1	1	1	1	1	1	1

可以看到如果按照文献[3]的判别方法,最后就将该全集划分为一个粒,这样的划分在理论上仍然是合理的。一个分布式数据库系统中,例如某一单元数据发生了异常,就如以上所假设的情况一样,t_1、t_2、t_3 和 t_4 进行了合并,并得到以上结果(如表 6 所示),又由于原始数据库中事务的数量远远大于 4,设还有事务为 $t_j(j>4)$,不失一般性设该事务存取矩阵 $tr_{ij}(j>4)$ 中,i 相同即为同一个事务中单元数据为 1 的数目大于为 0 的数目,如果在这种情况下继续按照文献[3]所介绍的方法,由于表 6 中描述的单元数据全为 1,在合并时只要是 $tr_{ij}(j>4)$ 中 1 的数目大于 0 的事务 t_j 就会合并,如果利用了该算法所产生的数据分片,将会直接导致全集划分过于简单,掩盖了一些站点上事务的细节,也会导致事务处理的失败。因此,如果因为原始数据库中某一数据单元发生异常,从而导致上述情况的发生,该算法产生的数据分片的结果必是不可用的。若一个单元数据发生异常,从而导致全集划分的失败,可以称这个异常数据为问题数据。问题数据出现的后果是严重的,所以在使用文献[3]的算法时,必须避免问题数据的产生。

4 问题解决方案探讨

由于文献[3]的算法也是基于数据本地化上进行的数据分片,所以,同一站点中的事务就有很多的相同之处,可以利用这一点来建立校验机制,排除一些异常的单元数据,减少问题数据的出现率。

下面介绍一下该校验机制。一个分布式数据库中的站点具有大量的事务,同时在同一站点中各个事务存储的单元数据有很大的相似之处。这种大量的数据且事务之间的相似性,使得可以利用古数学中“黄金比例”[9~11]的观点提供一个简单校验机制。按照古数学中的“黄金比例”的观点,设某一单元数据 r_i,有事务 $t_j(0<j<n)$,$tr_{ij}(i$ 固定,$0<j<n)$ 中,如果 tr_{ij} 等于 1

的数目与等于0的数目之比大于0.618或者小于0.382都可以认为是异常数据,系统可以提示数据库管理人员是否要进行检查。

文中将选取一个由以太网连接的具有100个站点的一个分布式系统,从中任意选取一个站点,将这一个站点的事务进行处理。首先将全集按地区进行排序,则修改后的数据单元中的各个元组基本上保证了都是同一地区的,在一定程度上保证了数据的本地性。一个站点中的事务总量为500,全集中的单元数据的总量为1000。对于一个固定的单元数据,详细地分析一下500个事务对它的访问情况。同时也是由于单元数据的数量比较大,因此也可以随即选择600个单元数据进行分析,分析以后得到数据,发现有78.17%的单元数据符合以下情况,$tr_{ij}(i$ 固定,$0<j<n)$ 等于1的数目与等于0的数目之比位于0.618与0.382之间。也就是,在所提供的500个事务中对这469个单元数据的访问情况,共有78.17%的单元数据符合"黄金分割点"的可信区间。由此可见,在一个正常的准确的数据库中,同一站点的事务对单元数据的访问大多数符合"黄金分割点"的规律。因此在算法的初期可以使用该方法对其进行检测,如果 $tr_{ij}(i$ 固定,$0<j<n)$ 等于1的数目与等于0的数目之比大于0.618或者小于0.382都可以认定数据库出现了异常需要检测。

同样文中还是选取同一站点中的500个事务,600个单元数据进行分析,如果随即篡改了一系列的数据,再计算 $tr_{ij}(i$ 固定,$0<j<n)$,对位于0.618与0.382区间之外的数据进行考察,会发现有总共篡改了223个单元数据中的事务访问情况,其中有169个单元数据的修改使 $tr_{ij}(i$ 固定,$0<j<n)$ 中的比值改变到了0.618与0.382区间之外,可以直接检测出。另外还有54组数据无法检测出,由此可见该校验机制有一定的效果。

由此可见,基于粒计算的数据分片算法只有在数据库理想的安全的情况下,才具有一定的实用价值。如果无法保证本地数据库的数据安全性,或者没有达到一定的安全级别,使用该分片算法,可能会导致分片结果的混乱和分片结果的不可应用。

5　结束语

文中通过对基于粒计算的数据分片算法的深入研究,得出该算法中过分的依赖于两个事务中值不相同的情况。如果数据库中的数据的存储遭到了篡改,该算法在实施过程中会出现大的误差,最终可直接导致数据分片的结果不可用。

文中利用古数学中"黄金分割"的思想而设定校验机制,利用该机制可以有效地避免一部分问题数据的出现。但是该方法无法保证其数据库的完全准确性,上述的情况还有可能发生。由文中的论述可知,要使得该算法有实际的应用效能,就必须要保证数据的准确性。因此就如何选择一个合理高效的校验机制,既能保证数据的可靠性又能保证算法的高效性,以及在什么时间进行校验从而保证算法的可靠性,是今后研究工作中的重点。

参考文献:

[1] Ghandeharizadeh S, Dewitt D. Hybrid-range partitioning strategy: A new declustering strategy for multiprocessor database machinese[C]//in: Proc. of the 16th VLDB Conf. Brisbance, Australia: [s. n.], 1990: 481-492.

[2] Navathe S, Ceri S, Wiederhold G, et al. Vertical Partitioning Algorithms for Database Design[J]. ACM Transactions on Database Systems, 1984, 9(4): 680-710.

[3] 吴润秀,吴水秀,刘　清. 基于粒计算的数据分片算法[J]. 计算机应用,2007,27(6):1388-1391.

[4] Pawlak Z. Rough set approach to multi-attribute decision analysis[J]. European Journal of Operational Research, 1994, 72(3): 443-459.

[5] 刘　清,孙　辉,王洪发. 粒计算研究现状及基于Rough逻辑语义的粒计算研究[J]. 计算机学报,2008,31(4):544-555.

[6] Banerjee M, Khan A. Propositional logic from rough set theory[C]//Transactions on Rough Sets VI. Lecture Notes in Computer Science 4374. Berlin: Springer, 2007: 1-25.

[7] 毛国君. 高级数据库原理与技术[M]. 北京:人民邮电出版社,2004.

[8] 包从剑,李从剑,施化吉. 可扩展和可交互的数据清洗系统[J]. 计算机技术与发展,2007,17(7):84-90.

[9] Liu Q. Granules and reasoning based on granular computing [C]//Lecture Notes in Artificial Intelligence 2718. Berlin: Springer-Verlag, 2003: 516-526.

[10] Liu Q, Jiang F, Deng D Y. Design and implement for the diagnosis software of bood viscosity syndrome based on morphology on Gr[C]//Lecture Notes in AI 2639. Berlin: Springer-Verlag, 2003: 413-420.

[11] Liu Qing. Granules and its research of decomposing & amalgamation[C]//Progress of Artificial Intelligence in China 2003-Proceeding of the 2003 National Conference on Artificial Intelligence(CAAI-10)(2/2). [s. l.]: [s. n.], 2003: 1411-1416.